

Design Doc: Trigger changed event when adding/removing virtual devices

This Document is Public

Authors: chfremer@chromium.org

One-page overview

Summary

The video capture service has the ability to add and remove [virtual video capture devices](#) that can be used to feed video streams from external processes into Chromium. Currently, adding or removing such virtual devices does not trigger an event. As a result, clients are forced to poll for changes. And even such polling may not work if device enumeration results are cached between the polling site and the video capture service.

The goal of this work is to make sure adding and removing virtual devices always emits an appropriate event to JavaScript clients, i.e. [MediaDevices.ondevicechange](#).

Platforms

All

Teams Involved

WebRTC-MTV team (point of contact: chfremer@chromium.org)

MRI-NYC team (lasoren@chromium.org, weigua@chromium.org)

Home team (zqiu@chromium.org)

Bug

<https://bugs.chromium.org/p/chromium/issues/detail?id=885256>

Code affected

Video capture service (services/video_capture/*)

Design Considerations

Monitoring of physical devices

Currently, monitoring of (physical, or at least OS-level) video capture devices being added and removed is handled globally via `base::SystemMonitor`. This includes device categories other than video capture. Longer-term it may make sense to move the responsibility of monitoring added and removed video capture devices to the video capture service, because that is where all other interactions with video capture devices are happening.

Devices added/removed while video capture service is not running

Since the video capture service is designed to be started on-demand and shut down when not being used, it may happen that devices are added/removed while the service is not running. However, this is only a concern for non-virtual devices, since at least one client must be connected in order to add or remove virtual devices, meaning the service must be running.

When does monitoring start and stop?

A potential solution to triggering device change events in the Browser process whenever a virtual device is added or removed to/from the video capture service would be to register the Browser process usage as an observer to the service, and raise an event to the observer when a device is added or removed. However, this solution turns out to be problematic, because the video capture service is designed to shut down when idle. An example for a problematic sequence of events would be the following:

1. Browser starts
2. Browser is asked to enumerate devices
3. Browser connects to service, and registers itself as observer for virtual device changes
4. Service shuts down, because no more usage, observer is disconnected
5. External process connects to service
6. External process adds virtual device
7. No event reaches the Browser or JavaScript apps, because it is not currently registered as an observer.

The same issue is present if an external process adds/removes virtual devices before the Browser ever connected to the service. Making matters worse is that there is no guaranteed way to make the Browser connect to the service from a Renderer/JavaScript app. Enumerating devices may not be sufficient, because there can be caching happening inside the Browser process, preventing a re-connect to the service.

Proposed Solution A

Proactively call the Browser process from the video capture service whenever we want to trigger a devices changed event. The downside of this is that we have to give the video capture service knowledge about the content_browser service. But this might be okay, since services are already abstracted through interfaces and manifests.

Note: This solution has been rejected, see <https://chromium-review.googlesource.com/c/chromium/src/+1237379> for details.

TODO: Consider what should happen if the video capture service shuts down while there are still virtual devices registered. One option would be to keep the service alive while there are virtual devices registered in order to prevent this from happening. Alternatively, we could emit a devices changed event before the service shutdown.

Proposed Solution B

Register class content::ServiceVideoCaptureProvider as a DeviceChangedObserver with the video_capture service and have it propagate events when virtual devices are added or removed. Since the video_capture service may be started by clients other than the browser process, register a ServiceManagerListener to listen for service starts and automatically register as DeviceChangedObserver when the service is started.

In order to mitigate race of an external client starting the service and adding virtual devices vs. the browser process seeing the service start and registering as DeviceChangedObserver, ask the service to immediately raise an event if virtual devices have already been present at time of registration.

Design Decisions

- Keep monitoring of physical devices at base::SystemMonitor for now.
- Use Proposed Solution B from the section above.

Implementation Plan

CL Title/Link	CL Description	Status
[Video Capture Service] Add observer API for adding/removing virtual devices	Add an API to the video capture service that allows observers to register and receive events whenever virtual devices are added or removed.	Landed

[Video Capture Service] Trigger MediaDevices.onddevicechange event when adding/removing virtual devices	Register class content::ServiceVideoCaptureProvider as a DeviceChangedObserver with the video_capture service and have it propagate events when virtual devices are added or removed. Since the video_capture service may be started by clients other than the browser process, register a ServiceManagerListener to listen for service starts and automatically register as DeviceChangedObserver when the service is started. In order to mitigate raciness of an external client starting the service and adding virtual devices vs. the browser process seeing the service start and registering as DeviceChangedObserver, ask the service to immediately raise an event if virtual devices have already been present at time of registration.	Under review
---	--	---------------------